Query: how to create and setup react project from github and configure it to run on subdomain

Result #1

```
if (!customElements.get('component-tag-name')) {



    customElements.define('component-tag-name', ComponentName);



}
```

4. Build the React Web Component

- Update the package.json with bundling command "build:webpack": "webpack --mode production"
- Use npm run build:webpack to generate a build of your React web component. The build folder will contain the necessary assets.

5. Use the Web Component in the Template Application

- After building the component, we can use it in any HTML page by referencing the built JavaScript file (e.g., main.js) in the <script> tag of the HTML file.
- We can then use the custom element (e.g., <component-tag-name></component-tag-name>) just like any other HTML element.

Source: **../Table-of-Contents\User-Interface-Framework\Solution-Strategy\Micro-frontend-Architecture\Web-Components\React-Web-Components.md**

Result #2

Overview

This section intends to document the steps required for creating web components in Angular

Steps to Create Web components in React

1. Set Up Your React Project

- Create a new React project
- Add the react-to-webcomponent package, which is necessary for web components, by running
  npm install react-to-webcomponent

2. Create & design the component

- Define the required component's template, logic, and styles in the newly created component files.

3. Expose React Component as Web Component

- In the index.js file, use createCustomElement to convert your React component into a custom element. This involves setting the component as an entry component and defining it as a custom element
  ```
   const ComponentName= reactToWebComponent(component, React, ReactDOM);
  if (!customElements.get('component-tag-name')) {
     customElements.define('component-tag-name', ComponentName);
  }
  ```

4. Build the React Web Component

Source: **../Table-of-Contents\User-Interface-Framework\Solution-Strategy\Micro-frontend-Architecture\Web-Components\React-Web-Components.md**

Result #3

Install vsts-npm-auth

npm i -g vsts-npm-auth

For more details refer:

https://dev.azure.com/Diatomic/UI%20Framework/_artifacts/feed/Diatomic/connect

1.4. Install Git

- Download the Git Installer as per your OS and install it to your system.
- Verify the installation with:
  git -v

2. Clone the existing repo to your system

git clone repo_url
Example
git clone https://dev.azure.com/Diatomic/UI%20Framework/_git/uob-shell-webapp

3. Install Dependencies

3.1 Move into the directory:

cd uob-shell-webapp

3.2 Configure Private NPM repository to access Widgets

Root Directory of the project contains a file named .npmrc
The .npmrc file is a configuration file used by npm (Node Package Manager) to manage various settings related to package management in a Node.js project.

Source: **../Table-of-Contents\Use-Case-Development%3A-Step%2Dby%2DStep-Guides\UI-Framework-Use-Case-Web-Client.md**

Result #4

📂 Example Repository Structure

plaintext

```
UseCase1 (ADO Project)
│── repo1 (sub-use case)
│   ├── azure-pipelines.yml (CI Pipeline)
│   ├── Dockerfile (if applicable)
│   ├── nginx.conf (if applicable)
│   ├── gitversion.yml (image versioning detailed in 1.3.4)
│── repo2 (sub-use case)
│   ├── azure-pipelines.yml (CI Pipeline)
│   ├── Dockerfile (if applicable)
│   ├── nginx.conf (if applicable)
│   ├── gitversion.yml (image versioning detailed in 1.3.4)
│── repo3 (sub-use case)
│   ├── azure-pipelines.yml (CI Pipeline)
│   ├── Dockerfile (if applicable)
│   ├── nginx.conf (if applicable)
│   ├── gitversion.yml (image versioning detailed in 1.3.4)
```

CI pipelines are defined at the repository level, calling a shared template based on the project’s technology stack.

Source: **../Table-of-Contents\Platform-Administration\DevOps-Administration.md**

Result #5

2. Azure DevOps Repository

To create an Azure DevOps repository, open an existing DevOps project and create the repository as shown in the official documentation of Microsoft here.

Once the repository has been created, it is expected to upload the application source code. Let's show a demo on the previously created project.

By default, when creating a new project, a new repository will be created with the same name, but we could choose to create another one if the existing is used by any other use case. For the purpose of this documentation, we will be using the default repository Documentation-Demo.

Next, a hello world solution on node.js will be pushed as part of the current demo. The source code is on a local machine and will be pushed to the Documentation-Demo repository.

Source: **../Table-of-Contents\Use-Case-Development%3A-Step%2Dby%2DStep-Guides\Build-a-Use-Case.md**

Result #6

Note: You don't need to do this every time. npm will give you a 401 Unauthorized error when you need to run it again.

3.3 Run Command to install dependencies:

npm install

This step will install all dependencies required to run the application.

4. Open the project in your preferred IDE

Open the project in your preferred IDE example VSCode and make required changes

5. Running the Shell App

Navigate to the uob-shell-webapp directory and start the application:

```
npm run start
```

Open your browser and visit:

```
http://localhost:4200/
```

Frequently Asked Questions (FAQ)

1. How to add IAM integration in the base application ?

Steps to integrate IAM (Identity and Access Management) and secure an Angular application can be found in the following resources:
Login Page
Security

Source: **../Table-of-Contents\Use-Case-Development%3A-Step%2Dby%2DStep-Guides\UI-Framework-Use-Case-Web-Client.md**